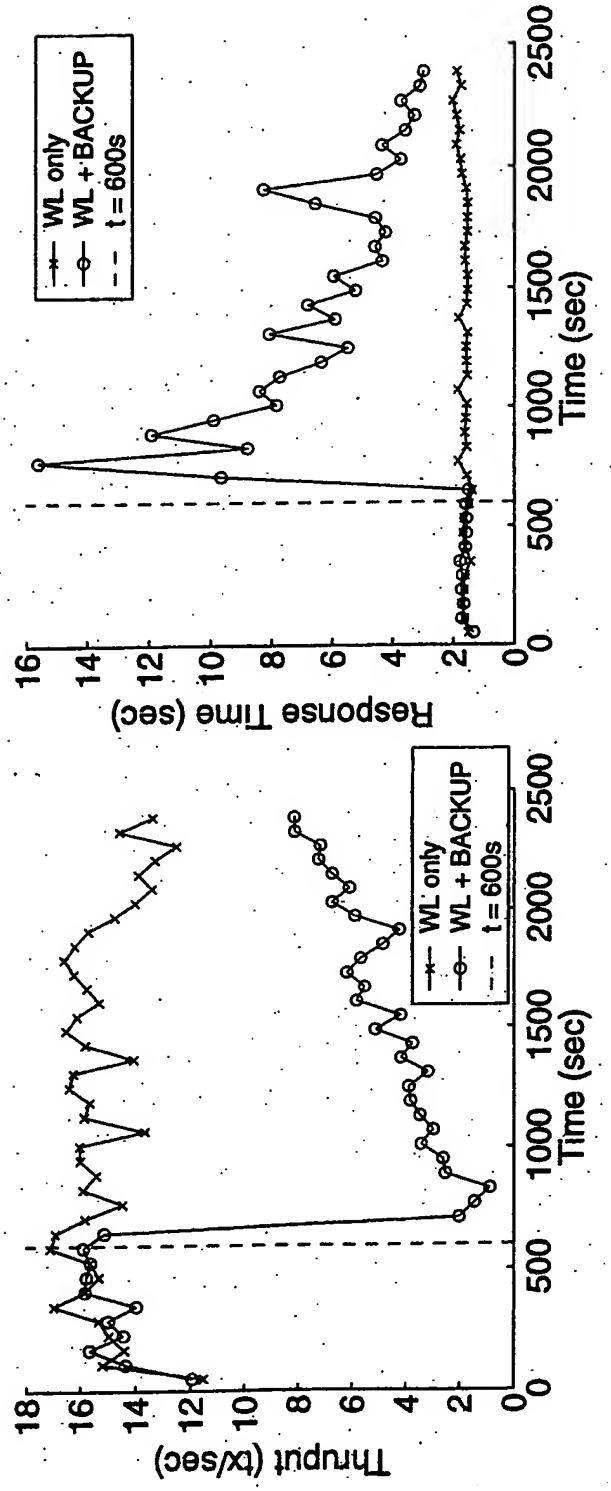
Response Time (sec)
Time (sec)
WL only
WL + BACKUP
t = 600s
Thruput (tx/sec)
Time (sec)
WL only
WL + BACKUP
t = 600s

Fig. 1

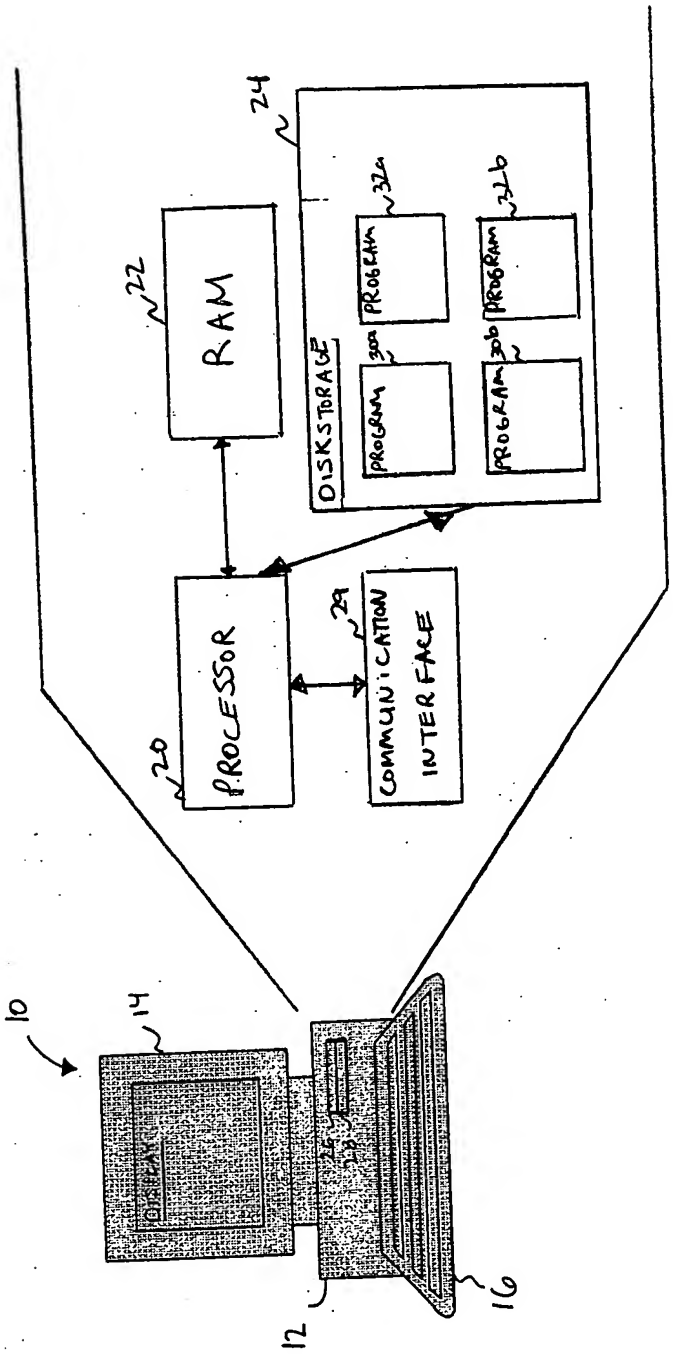
10
14
DISPLAY
12
16
PROCESSOR
20
RAM
22
DISK STORAGE
24
PROGRAM
32a
PROGRAM
32b
PROGRAM
30a
PROGRAM
30b
COMMUNICATION INTERFACE
29

Fig. 2

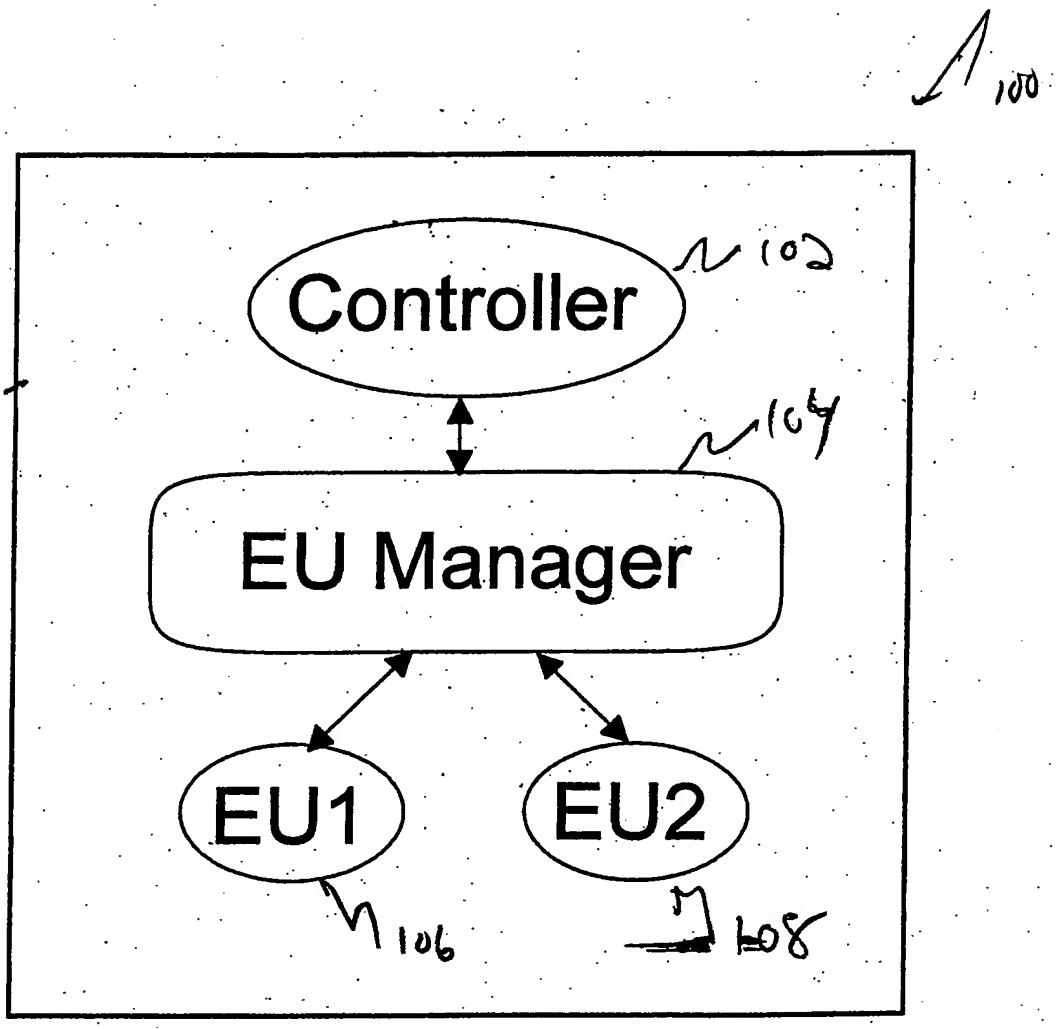

Fig. 3

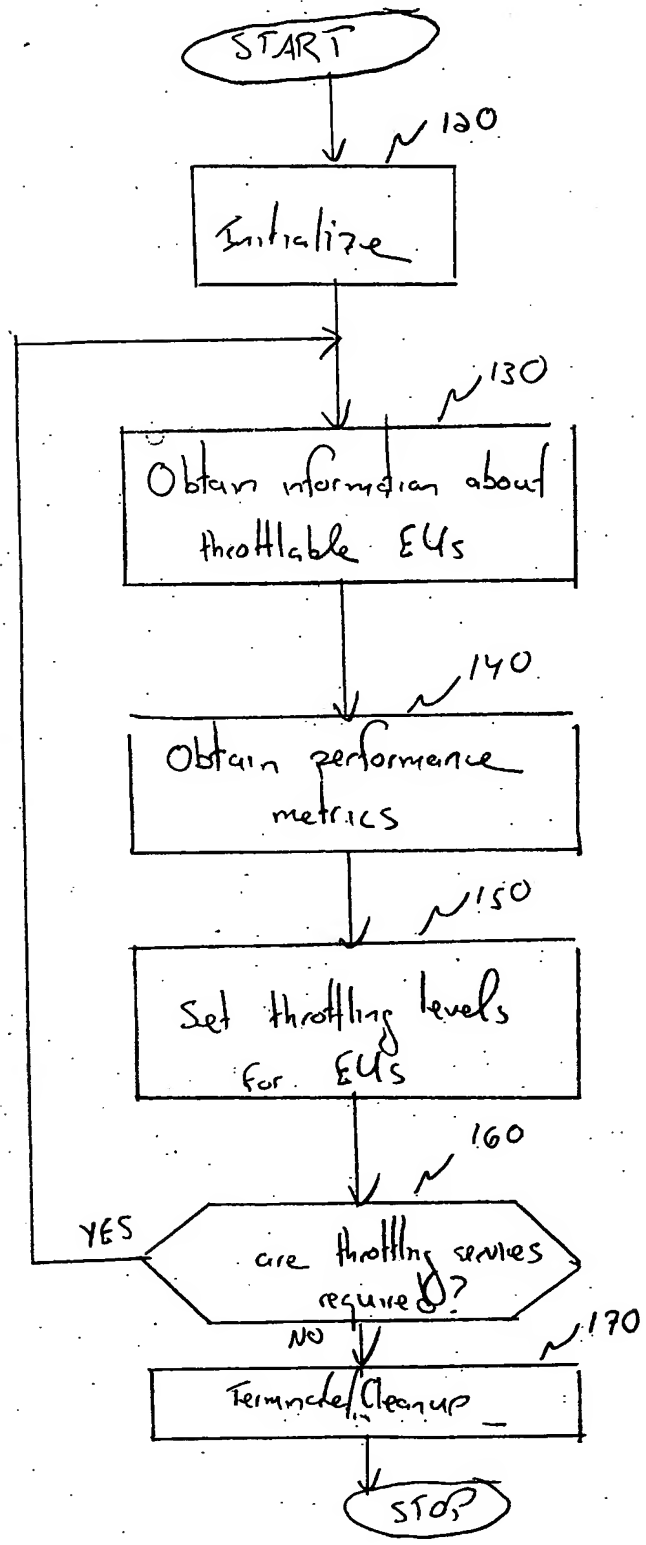
START
120
Initialize
130
Obtain information about throttlable EUs
140
Obtain performance metrics
150
Set throttling levels for EUs
160
YES
are throttling services required?
NO
170
Terminate/Cleanup
STOP

Fig.4

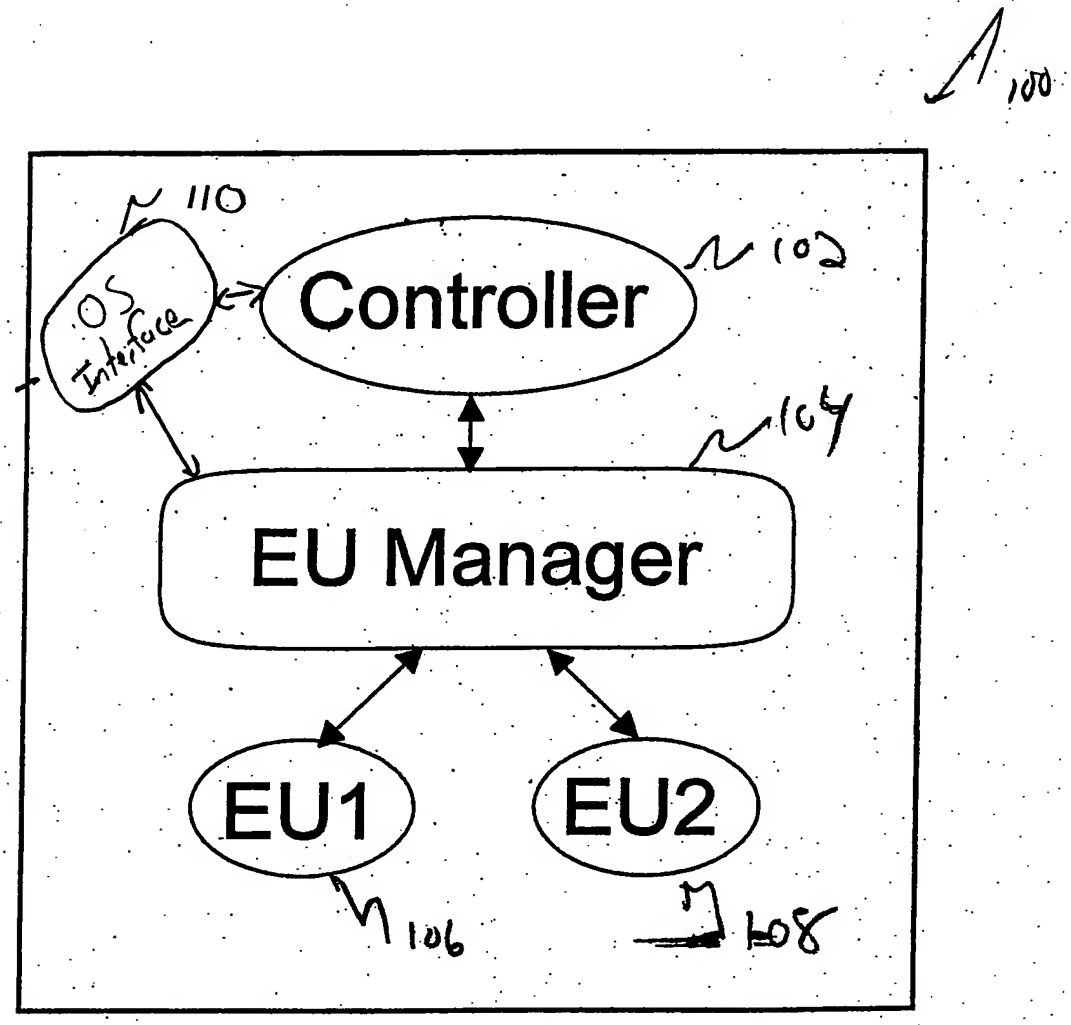
100
110
OS Interface
Controller
102
104
EU Manager
EU1
EU2
106
108

Fig. 5

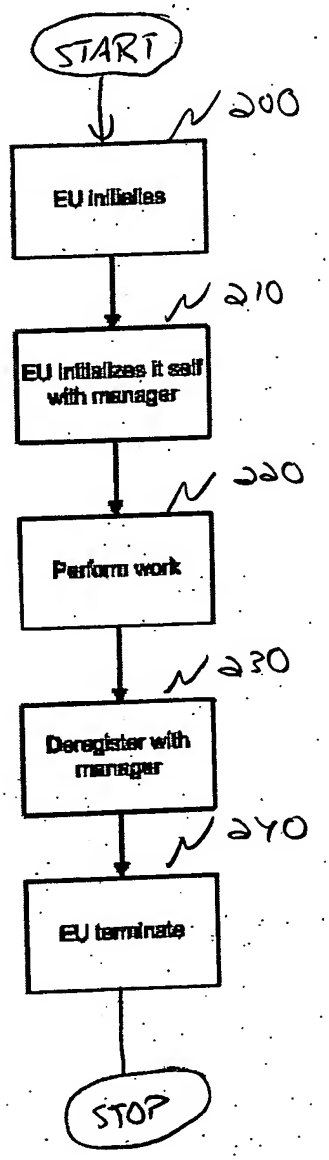
START
200
EU initialies
210
EU initializes it self with manager
220
Perform work
230
Deregister with manager
240
EU terminate
STOP

Fig. 6

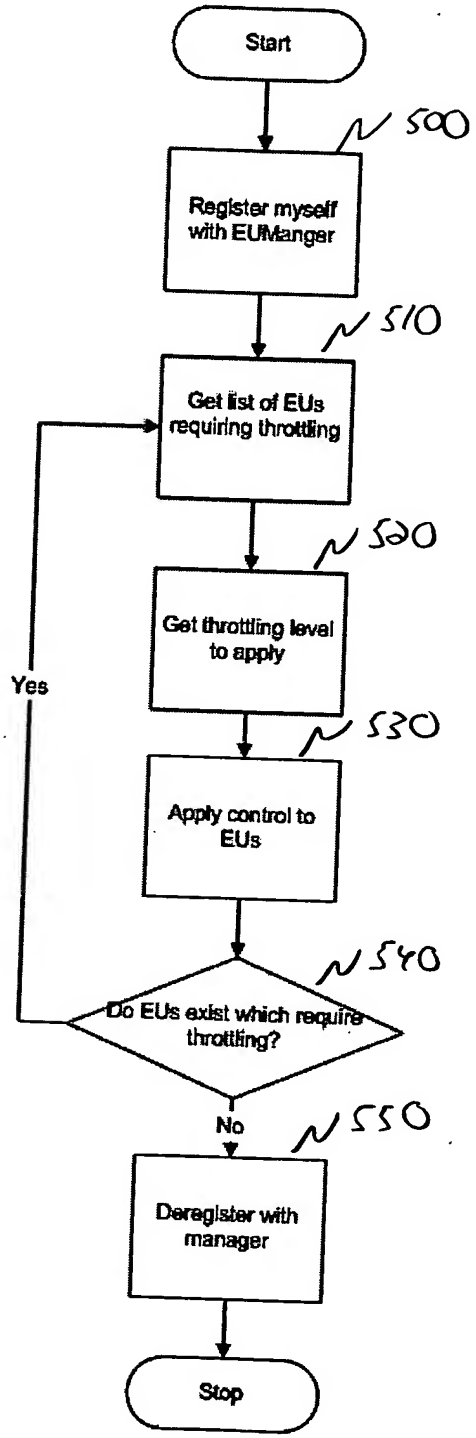
Start
500
Register myself with EUManger
510
Get list of EUs requiring throttling
520
Get throttling level to apply
530
Apply control to EUs
540
Do EUs exist which require throttling?
Yes
No
550
Deregister with manager
Stop

Fig. 6b

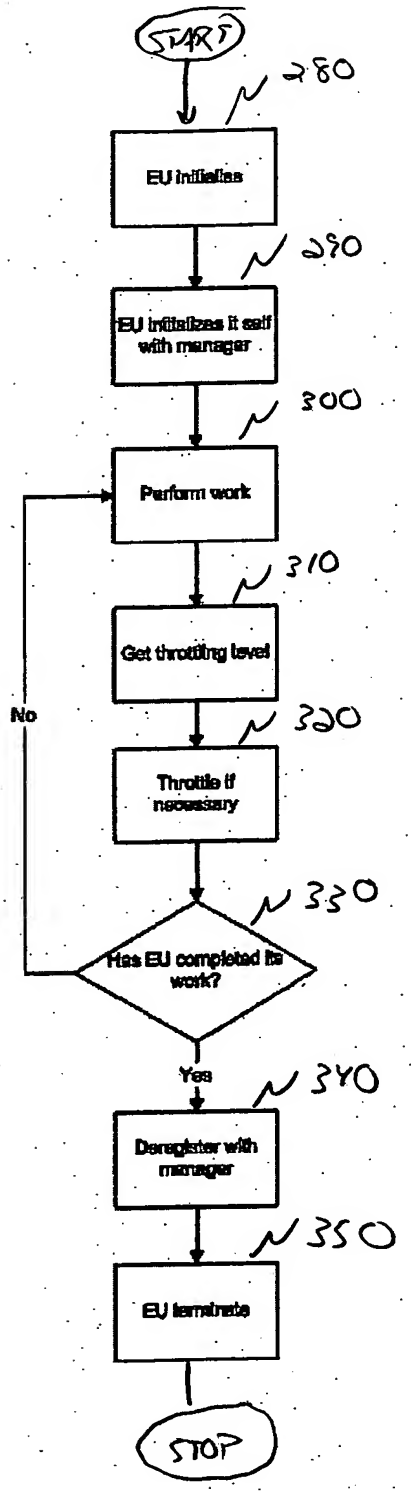
START
280
EU initialize
290
EU initializes it self with manager
300
Perform work
310
Get throttling level
320
Throttle if necessary
330
Has EU completed its work?
No
Yes
340
Deregister with manager
350
EU terminate
STOP

Fig. 7

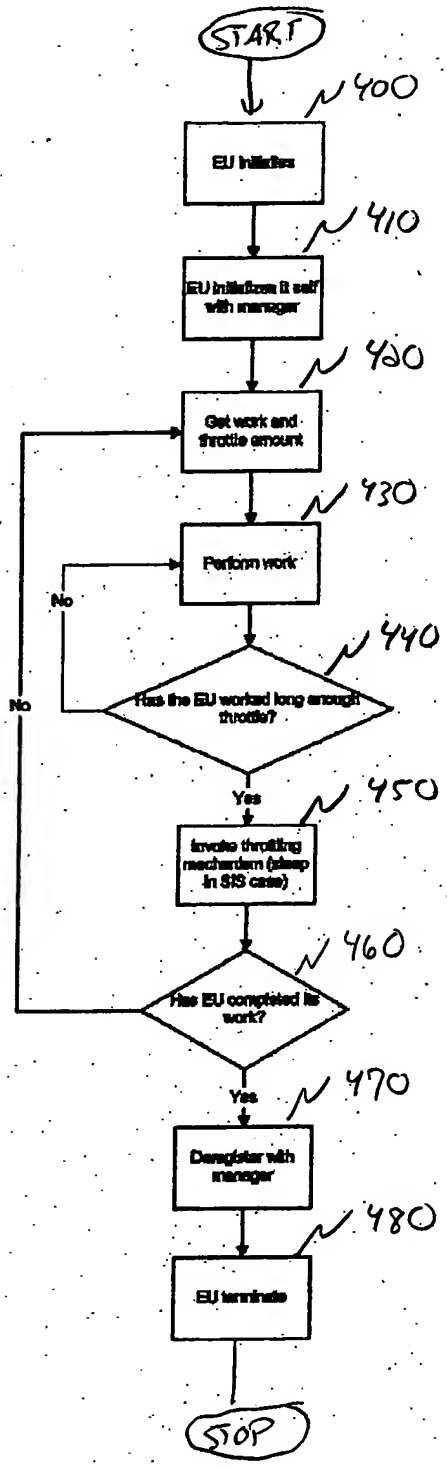
START
400
EU initializes
410
EU initializes it self with manager
420
Get work and throttle amount
430
Perform work
No
440
Has the EU worked long enough throttle?
No
Yes
450
Invoke throttling mechanism (sleep in SIS case)
460
Has EU completed its work?
Yes
470
Deregister with manager
480
EU terminate
STOP

Fig. 8

```
COMPONENT ThrottleAgent
BEGIN
EUManager->RegisterThrottleAgent() ;
While (EUManager->IsThrottlingRequired())
BEGIN
EUList = EUManager->GetListOfEUs( self ) ;
ThrottlingLevels = EUManager->GetThrottlingLevel( EUList ) ;
... apply ThrottlingLevel to EUs in EUList ...
END
EUManager->DeregisterThrottleAgent() ;
END
```

Fig. 9

```
COMPONENT EU
BEGIN
... Initialization ...
EUManager->RegisterEU(...args...) ;

While( NOT Done )
BEGIN
... Do some work ...
ThrottlingLevel = EUManager->GetThrottlingLevel( self ) ;
Throttle(ThrottlingLevel) ;
END

... Termination ...
EUManager->DeRegisterEU() ;
END
```

Fig. 10

```
FUNCTION Utility()
BEGIN
     WHILE (NOT done)
     BEGIN
          ... do some work ...
          ThrottleIfNeeded()
     END
END
```

(a) Inserting SIS point

```
FUNCTION ThrottleIfNeeded()
BEGIN
     (workTime, sleepTime) = GetThrottlingLevel() ;
     timeWorked = Now() - workStart ;
     IF (timeWorked > workTime)
          SLEEP( sleepTime ) ;
          workStart = Now() ;
     ENDIF
END
```

(b) SIS implementation

Fig. 11

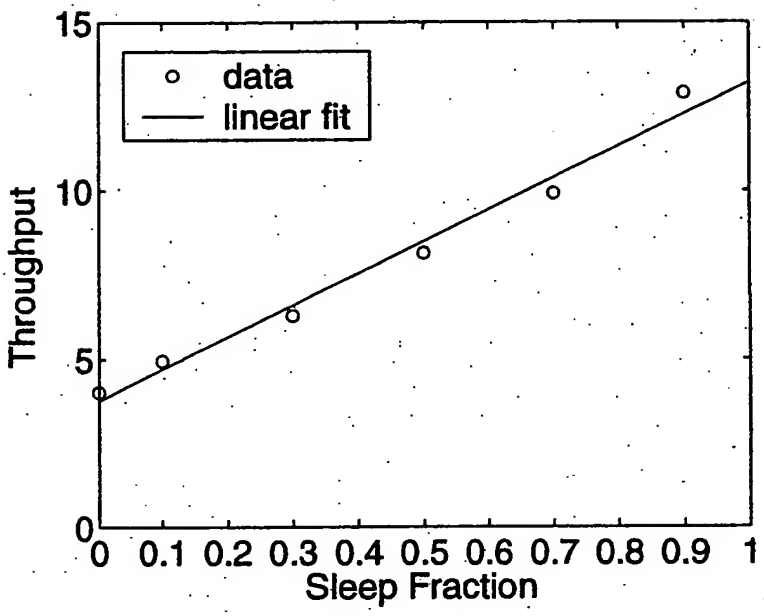
15
10
5
0
Throughput
data
linear fit
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Sleep Fraction

Fig. 12

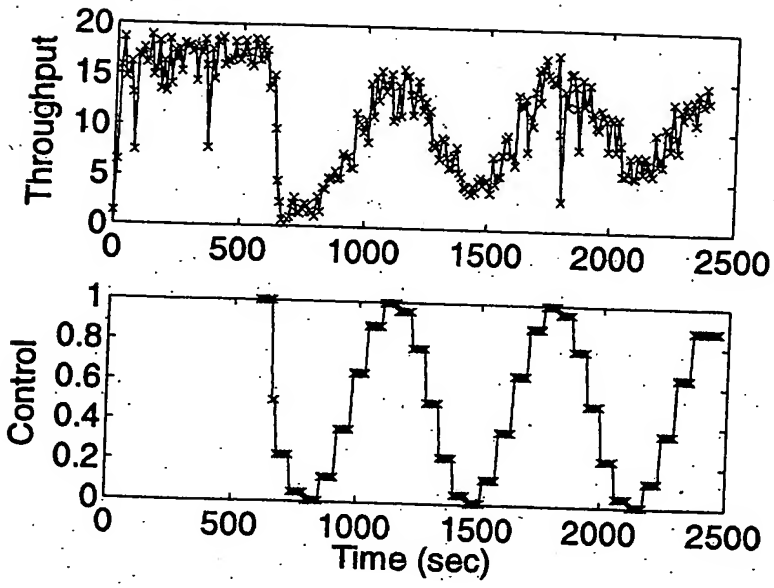
Throughput
20
15
10
5
0
0
500
1000
1500
2000
2500
Control
1
0.8
0.6
0.4
0.2
0
0
500
1000
1500
2000
2500
Time (sec)

Fig. 13